# PATENT ABSTRACTS OF JAPAN

(11)Publication number : 2001-243066
(43)Date of publication of application : 07.09.2001

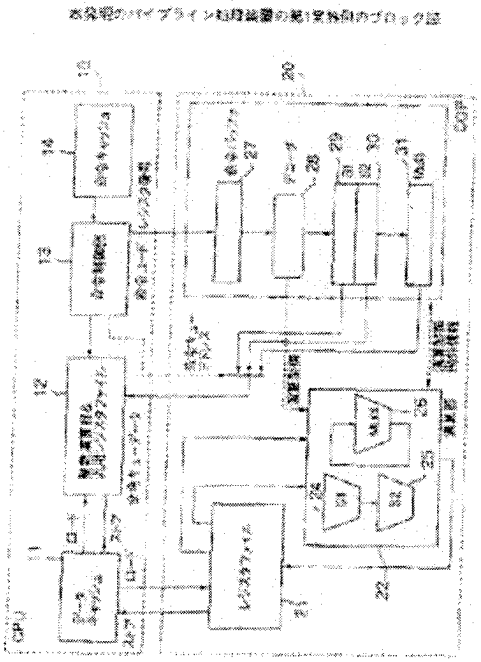
(51)Int.Cl. G06F 9/38

(21)Application number : 2000-054832
(22)Date of filing : 29.02.2000

(71)Applicant : FUJITSU LTD
(72)Inventor : TSUJI MASAYUKI

## (54) PIPE LINE PROCESSING METHOD AND PIPE LINE PROCESSOR USING THE METHOD

(57)Abstract:
PROBLEM TO BE SOLVED: To provide a pipe line processing method by which the deterioration of instruction processing performance is prevented and power consumption and costs are reduced and a pipe line processor using the method.
SOLUTION: This pipe line processor is provided with first storage means 29 and 30 for storing arithmetic instructions supplied from a central arithmetic unit to arithmetic units, first arithmetic units 24 and 25 for performing arithmetic operations according to the arithmetic instructions stored in the first storage means, a second storage means 31 for storing the arithmetic instruction when the arithmetic operation performed according to the arithmetic instruction requires a long time until the end of the arithmetic operation, and a second arithmetic unit 26 for performing the arithmetic operation according to the arithmetic instruction stored in the second storage means.



(19)日本国特許庁（ＪＰ）　(12)公開特許公報（A）　(11)特許出願公開番号
特開2001－243066
(P2001－243066A)
(43)公開日　平成13年９月７日(2001.9.7)

| (51)Int.Cl.⁷ | 識別記号 | ＦＩ | ﾃｰﾏｺｰﾄﾞ(参考) |
|---|---|---|---|
| Ｇ０６Ｆ　9/38 | ３１０ | Ｃ０６Ｆ　9/38 | ３１０Ｆ　５Ｂ０１３ |
| | | | ３１０Ｊ |
| | ３７０ | | ３７０Ｃ |

審査請求　未請求　請求項の数６　ＯＬ　（全９頁）

(21)出願番号　特願2000－54832(P2000－54832)

(22)出願日　平成12年２月29日(2000.2.29)

(71)出願人　000005223
富士通株式会社
神奈川県川崎市中原区上小田中４丁目１番１号

(72)発明者　辻　雅之
神奈川県川崎市中原区上小田中４丁目１番１号　富士通株式会社内

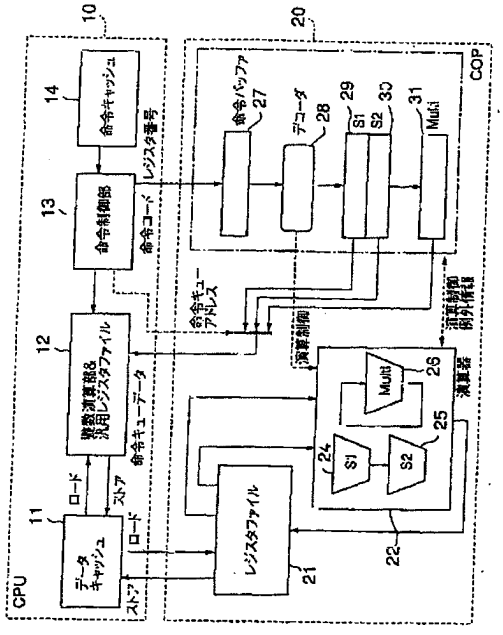
(74)代理人　100070150
弁理士　伊東　忠彦

Ｆターム(参考)　5B013 AA12 AA18 DD03

(54)【発明の名称】　パイプライン処理方法並びにその方法を利用するパイプライン処理装置

(57)【要約】
【課題】　命令処理性能の低下を防ぐことができ、消費電力及びコストを削減することが可能なパイプライン処理方法並びにその方法を利用するパイプライン処理装置を提供することを目的とする。
【解決手段】　中央演算装置から演算装置に供給される演算命令を格納する第１格納手段２９，３０と、第１格納手段に格納された演算命令に従って演算を行なう第１演算器２４，２５と、演算を行なった演算命令が演算終了までに時間の掛かる演算命令であれば、その演算命令を格納する第２格納手段３１と、第２格納手段に格納された演算命令に従って演算終了まで演算を行なう第２演算器２６とを有することにより上記課題を解決する。

本発明のパイプライン処理装置の第1実施例のブロック図



【特許請求の範囲】
【請求項１】　中央演算装置と演算装置とを相互接続して演算処理を行なうパイプライン処理方法において、
前記演算装置に供給される演算命令を格納する段階と、
前記格納された演算命令に従って演算を行い、演算終了までに時間の掛かる演算命令であるか確認する段階と、
演算終了までに時間の掛かる演算命令であれば、その演算命令を専用の格納場所にシフトする段階と、
前記専用の格納場所にシフトした前記演算命令の演算を演算終了まで行なう段階とを有するパイプライン処理方法。
【請求項２】　演算終了までに時間の掛かる演算命令でなければ、その演算命令の演算結果を順次出力する段階を更に有する請求項１記載のパイプライン処理方法。
【請求項３】　中央演算装置と演算装置とを相互接続して演算処理を行なうパイプライン処理装置において、
前記演算装置に供給される演算命令を格納する第１格納手段と、
前記第１格納手段に格納された演算命令に従って演算を行なう第１演算器と、
前記演算を行なった演算命令が演算終了までに時間の掛かる演算命令であれば、その演算命令を格納する第２格納手段と、
前記第２格納手段に格納された演算命令に従って演算終了まで演算を行なう第２演算器とを有するパイプライン処理装置。
【請求項４】　中央演算装置と演算装置とを相互接続して演算処理を行なうパイプライン処理装置において、
前記演算装置に供給される演算命令を格納する第１格納手段と、
前記第１格納手段に格納された演算命令に従って演算を行なう第１演算器と、
前記演算を行なった演算命令が演算終了までに時間の掛かる演算命令であれば、その演算命令を格納する複数の第２格納手段と、
前記複数の第２格納手段に格納されている演算命令の発行順序を示す識別手段と、
前記識別手段に従って最先に発行された演算命令を選択し、その演算命令に従って演算終了まで演算を行なう複数の第２演算器とを有するパイプライン処理装置。
【請求項５】　中央演算装置と複数の演算装置とを相互接続して演算処理を行なうパイプライン処理装置において、
前記複数の演算装置毎に、前記演算装置に供給される演算命令を格納する第１格納手段と、
前記第１格納手段に格納された演算命令に従って演算を行なう第１演算器と、
前記演算を行なった演算命令が演算終了までに時間の掛かる演算命令であれば、その演算命令を格納する第２格納手段と、
前記第２格納手段に格納されている演算命令の発行順序を示す識別手段と、
前記識別手段に従って最先に発行された演算命令を選択し、その演算命令に従って演算終了まで演算を行なう第２演算器とを有し、
前記複数の演算装置の識別手段に優先順位を設定しておくことを特徴とするパイプライン処理装置。
【請求項６】　前記演算終了までに時間の掛かる演算命令は、演算終了までに複数サイクルの時間を必要とするマルチプル演算命令であることを特徴とする請求項３乃至５何れか一項記載のパイプライン処理装置。
【発明の詳細な説明】
【０００１】
【発明の属する技術分野】本発明は、パイプライン処理方法並びにその方法を利用するパイプライン処理装置に係り、特に、中央演算装置（Ｃｅｎｔｒａｌ　Ｐｒｏｃｅｓｓｉｎｇ　Ｕｎｉｔ）と演算装置とを相互接続して非同期に演算を行なうパイプライン処理方法並びにその方法を利用するパイプライン処理装置に関する。
【０００２】
【従来の技術】近年、コンピュータ装置は処理速度の更なる高速化が要求されるようになり、中央演算装置（以下、ＣＰＵという。）単体では要求される処理速度を満たさない場合が生じている。そこで、高速演算を実行する為の演算装置を別途設け、ＣＰＵと並列に非同期演算を行なわせることによりＣＰＵを補完する処理方法が用いられるようになってきている。このような演算装置としては、例えば浮動小数点演算を行なうコプロセッサ（Ｃｏ－Ｐｒｏｃｅｓｓｏｒ：以下、ＣＯＰという。）がある。
【０００３】なお、パイプライン処理方法とは、命令の処理過程を複数の段階に分割し、複数個の命令をパイプライン的に先回り並行処理する制御方法である。パイプライン処理方法は、各段階の時間間隔毎に命令実行が可能であり、単位時間当りの処理能力が向上する。
【０００４】図１は、パイプライン処理装置の一例のブロック図を示す。パイプライン処理装置は、ＣＰＵ１００とＣＯＰ２００とを含む構成である。ＣＰＵ１００はＣＯＰ２００と相互接続されており、浮動小数点演算等のＣＯＰ２００を使用する演算命令があると、その演算命令の命令コードとレジスタ番号とをＣＯＰ２００に供給する。
【０００５】ＣＯＰ２００はＣＰＵ１００から供給される命令コード及びレジスタ番号を命令バッファ２３０に格納する。命令バッファ２３０に格納された演算命令は各種パイプラインハザードが解消された段階でパイプラインド演算器２２０で実行される。演算命令は、パイプラインド演算器２２０の演算ステージＳ１，Ｓ２等に対応するように命令キュー２４０，２４１を伝播していく。

【0006】最終の演算ステージS2では例外チェックが行われ、演算が正常終了しているか判定される。演算が正常終了していれば、演算命令は命令キュー241からデキューされる。また、演算結果はパイプラインド演算器220からレジスタファイル210に格納される。演算が正常終了せず例外が検出された場合、演算命令は命令キュー241に残る。そして、例外情報が命令キュー241に登録されると共に、CPU100に対して割り込み要求が行われる。このとき、命令キュー240にある後続の命令は、未完了の命令として命令キュー240に格納される。

【0007】但し、マルチサイクルを必要とするマルチプル演算命令の場合、演算レイテンシが長い為にマルチプル演算命令の情報が、ある命令キュー240，241に複数サイクルの間留まることになる。この間、後続の演算命令は前のステージの命令キュー240又は命令バッファ230で待ちの状態となる。この待ちの状態を最小限にする為、命令バッファ230は待ち命令キュー231，待ち命令キュー232を設けて複数段の構成とし、CPU100が供給する演算命令を格納する。このように、従来のパイプライン処理装置は演算命令と演算の実行との対応を容易とし、例外検出時の割り込み処理も容易としていた。

【0008】

【発明が解決しようとする課題】図2は、パイプライン処理装置の一例のタイムチャートを示す。図2のタイムチャートは、演算命令がマルチプル演算命令a，パイプラインド演算命令b，パイプラインド演算命令c，パイプラインド演算命令d，パイプラインド演算命令eの順番で実行される場合について示したものである。

【0009】まず、時刻tにマルチプル演算命令aがCPU100に供給され、命令バッファ230を介して命令キュー240に格納される。しかし、マルチプル演算命令aは実行の終了までにマルチサイクルを必要とするので、時刻t+2から命令キュー240に留まり続ける。

【0010】続いて時刻t+1にはパイプラインド演算命令bがCPU100に供給され、命令バッファ230に格納される。時刻t+3では、命令キュー240にマルチプル演算命令aが格納されているのでパイプラインド演算命令bが命令バッファ230から待ち命令キュー231に供給される。また、時刻t+4では、パイプラインド演算命令bが待ち命令キュー231から待ち命令キュー232に供給された後、待ち命令キュー232に留まり続ける。

【0011】時刻t+2には、パイプラインド演算命令cがCPU100に供給され、命令バッファ230に格納される。時刻t+4では、命令キュー240にマルチプル演算命令aが格納されているのでパイプラインド演算命令cが命令バッファ230から待ち命令キュー231に供給された後、待ち命令キュー231に留まり続ける。

【0012】さらに、時刻t+3にはパイプラインド演算命令dがCPU100に供給され、命令バッファ230に格納される。しかし、待ち命令キュー231，232にパイプラインド演算命令b，cが留まり続けている為、パイプラインド演算命令dは命令バッファ230に留まり続ける。

【0013】従って、時刻t+4にパイプラインド演算命令eがCPU100に供給されても命令バッファ230に空きが生じない為、パイプラインド演算命令eは処理待ち状態、言い換えればCPUストール状態となる。つまり、従来のパイプライン処理装置は、マルチプル演算命令が供給されると後続の命令がマルチプル演算命令の完了待ちを行なう為、全体の命令処理性能が低下するという問題があった。また、実装されている待ち命令キュー231，232の数を増加すればCPUストール状態を減少させることができるが、消費電力の増加，コストの上昇等の問題があった。

【0014】本発明は、上記の点に鑑みなされたもので、命令処理性能の低下を防ぐことができ、消費電力及びコストを削減することが可能なパイプライン処理方法並びにその方法を利用するパイプライン処理装置を提供することを目的とする。

【0015】

【課題を解決するための手段】そこで、上記課題を解決するため、請求項1記載のパイプライン処理方法は、前記演算装置に供給される演算命令を格納する段階と、前記格納された演算命令に従って演算を行い、演算終了までに時間の掛かる演算命令であるか確認する段階と、演算終了までに時間の掛かる演算命令であれば、その演算命令を専用の格納場所にシフトする段階と、前記専用の格納場所にシフトした前記演算命令の演算を演算終了まで行なう段階とを有することを特徴とする。

【0016】このように、演算終了までに時間の掛かる例えばマルチプル演算命令を実行する場合に、そのマルチプル演算命令を専用の格納場所にシフトする段階を有することにより、後続の演算命令に対する命令処理性能の低下を防ぎ、命令バッファの段数を削減することにより消費電力及びコストの削減が可能である。

【0017】例えば、命令のアウトオブオーダ（Out-Of-Order）完了を許すアーキテクチャでは、必ずしも命令を発行順で実行完了させなくともよい。この場合、本願発明のような制御も可能となる。

【0018】また、請求項2記載のパイプライン処理方法は、演算終了までに時間の掛かる演算命令でなければ、その演算命令の演算結果を順次出力する段階を更に有することを特徴とする。

【0019】このように、演算終了までに時間の掛かる例えばマルチプル演算命令であっても他の演算命令と同

様のタイミングで格納場所を遷移させることができる為、後続の他の演算命令をストールさせることなく演算処理を行なうことが可能である。

【００２０】また、請求項３記載のパイプライン処理装置は、前記演算装置に供給される演算命令を格納する第１格納手段（例えば、図３における命令キュー２９，３０）と、前記第１格納手段に格納された演算命令に従って演算を行なう第１演算器（例えば、図３における演算ステージ２４，２５）と、前記演算を行なった演算命令が演算終了までに時間の掛かる演算命令であれば、その演算命令を格納する第２格納手段（例えば、図３における命令キュー３１）と、前記第２格納手段に格納された演算命令に従って演算終了まで演算を行なう第２演算器（例えば、図３における演算ステージ２６）とを有することを特徴とする。

【００２１】このように、演算終了までに時間の掛かる例えばマルチプル演算命令を実行する場合に、そのマルチプル演算命令を格納する第２格納手段とそのマルチプル演算命令を行なう第２演算器とを有することにより、後続の演算命令に対する命令処理性能の低下を防ぎ、命令バッファの段数を削減することにより消費電力及びコストの削減が可能である。

【００２２】また、請求項４記載のパイプライン処理装置は、前記演算装置に供給される演算命令を格納する第１格納手段と、前記第１格納手段に格納された演算命令に従って演算を行なう第１演算器と、前記演算を行なった演算命令が演算終了までに時間の掛かる演算命令であれば、その演算命令を格納する複数の第２格納手段（例えば、図５における命令キュー３７，３８）と、前記複数の第２格納手段に格納されている演算命令の発行順序を示す識別手段（例えば、図５におけるアドレス操作ビット３９，４０）と、前記識別手段に従って最先に発行された演算命令を選択し、その演算命令に従って演算終了まで演算を行なう複数の第２演算器（例えば、図５における演算ステージ３５，３６）とを有することを特徴とする。

【００２３】このように、複数の第２格納手段に格納されている演算命令の発行順序を示す識別手段を有することにより、演算命令の命令発行順に演算終了までに時間の掛かる例えばマルチプル演算命令を実行することが可能となる。

【００２４】また、請求項５記載のパイプライン処理装置は、複数の演算装置（例えば、図６におけるＣＯＰ５０，６０）毎に、前記演算装置に供給される演算命令を格納する第１格納手段と、前記第１格納手段に格納された演算命令に従って演算を行なう第１演算器と、前記演算を行なった演算命令が演算終了までに時間の掛かる演算命令であれば、その演算命令を格納する第２格納手段と、前記第２格納手段に格納されている演算命令の発行順序を示す識別手段と、前記識別手段に従って最先に発行された演算命令を選択し、その演算命令に従って演算終了まで演算を行なう第２演算器とを有し、前記複数の演算装置の識別手段に優先順位を設定しておくことを特徴とする。

【００２５】このように、複数の演算装置の識別手段に優先順位を設定しておくことにより、例えば異なる演算装置に同時にマルチプル演算命令が発行した場合に対応することができる。

【００２６】また、請求項６記載のパイプライン処理装置は、前記演算終了までに時間の掛かる演算命令は、演算終了までに複数サイクルの時間を必要とするマルチプル演算命令であることを特徴とする。

【００２７】このように、マルチプル演算命令を実行する場合に、後続の例えばパイプライン演算命令に対する命令処理性能の低下を防ぐことが可能であり、命令バッファの段数を削減することにより消費電力及びコストの削減が可能である。

【００２８】なお、上記括弧内の符号は理解を容易とするために附したものであり、一例にすぎない。

【００２９】

【発明の実施の形態】次に、本発明の実施の形態について図面に基づいて説明する。図３は、本発明のパイプライン処理装置の第１実施例のブロック図を示す。パイプライン処理装置は、相互接続されているＣＰＵ１０とＣＯＰ２０とを含むように構成される。また、ＣＰＵ１０は、データキャッシュ１１，整数演算部&汎用レジスタファイル１２，命令制御部１３，及び命令キャッシュ１４を含むように構成される。ＣＯＰ２０はレジスタファイル２１、演算器２２、命令バッファ２７、デコーダ２８、命令キュー２９，３０、及びマルチプル演算命令用の命令キュー３１を含むように構成される。

【００３０】ＣＰＵ１０の命令キャッシュ１４はプログラムが格納されており、演算命令を命令制御部１３に供給する。命令制御部１３は演算命令が供給されると、その演算命令が浮動小数点演算等のＣＯＰ２０を使用する演算命令であるか判断する。ＣＯＰ２０を使用する演算命令であると判断すると、その演算命令の命令コードとレジスタ番号とをＣＯＰ２０の命令バッファ２７に供給する。一方、ＣＯＰ２０を使用しない整数演算等の演算命令であると判断すると、その演算命令の命令コードとレジスタ番号とを整数演算部&汎用レジスタファイル１２に供給する。

【００３１】整数演算部&汎用レジスタファイル１２はレジスタ番号に従ってデータキャッシュ１１からデータを読出し、命令コードに従ってそのデータの演算を行なう。その後、整数演算部&汎用レジスタファイル１２は演算結果をデータキャッシュ１１に書込む。

【００３２】命令バッファ２７は命令制御部１３から供給された命令コードとレジスタ番号とを、各種パイプラインハザードが解消された段階でデコーダ２８に供給す

る。即ち、レジスタ干渉，ハードウェアの資源競合がないことを確認する。デコーダ２８は供給された命令コードをデコードし、命令キュー２９に演算命令を格納すると共に、演算命令及びレジスタ番号を演算器２２の演算ステージ２４に供給する。なお、各種パイプラインハザードが解消されていない場合、命令バッファ２７は命令コード及びレジスタ番号をデコーダ２８に供給せず、次サイクルで再度各種パイプラインハザードが解消されているか確認される。

【００３３】命令キュー２９は、格納されている演算命令をパイプライン形式で演算キュー３０に供給する。このとき、演算器２２の演算ステージ２４に格納されている演算命令及びレジスタ番号は、演算ステージ２５に供給される。演算ステージ２４から演算命令及びレジスタ番号が供給されると、演算ステージ２５は必要なデータをレジスタファイル２１から読出し、その演算命令に従って演算を行なう。

【００３４】つまり、演算ステージ２５に演算命令及びレジスタ番号が供給されると、次サイクルで演算結果が求まる。演算結果が求まると演算ステージ２５では演算例外の有無を調べ、演算が正常終了している場合は演算命令がデキューされる。そして、演算器２２からレジスタファイル２１に演算結果を供給する。一方、演算例外が発生している場合、演算ステージ２５及び命令キュー３０に演算命令及び演算例外情報を格納して割り込み動作に入る。

【００３５】マルチプル演算命令の場合、更に演算が続行するので演算ステージ２５及び命令キュー３０に格納されている演算命令及び演算例外情報がマルチプル演算命令用の演算ステージ２６及び命令キュー３１に遷移される。なお、演算の開始時に分かる演算例外、例えばゼロ除算は他のパイプラインド演算命令と同様に、例外検出を行なうことも可能である。

【００３６】マルチプル演算命令用の演算ステージ２６及び命令キュー３１に演算終了まで保持される演算命令は、演算終了時に再度演算例外の有無を調べられる。演算例外がなければ、演算ステージ２６及び命令キュー３１から演算命令がデキューされる。一方、演算例外があれば演算ステージ２６及び命令キュー３１に演算命令を残し、割り込み動作に入る。なお、演算結果はレジスタファイル２１に格納される。

【００３７】図3のパイプライン処理装置の処理について図4を参照しつつ説明する。図4は、パイプライン処理装置の一例のタイムチャートを示す。なお、図4では説明に不要な部分を省略している。図4のタイムチャートは、演算命令がマルチプル演算命令ａ，パイプラインド演算命令ｂ，パイプラインド演算命令ｃ，パイプラインド演算命令ｄ，パイプラインド演算命令ｅの順番で実行される場合について示したものである。

【００３８】まず、時刻ｔにマルチプル演算命令ａが命令キャッシュ１４から命令制御部１３に供給される。時刻ｔ＋１に命令制御部１３はマルチプル演算命令ａを命令バッファ２７に供給する。また、パイプラインド演算命令ｂが命令キャッシュ１４から命令制御部１３に供給される。

【００３９】続いて時刻ｔ＋２にマルチプル演算命令ａが命令バッファ２７から命令キュー２９に供給される。命令制御部１３はパイプラインド演算命令ｂを命令バッファ２７に供給する。また、パイプラインド演算命令ｃが命令キャッシュ１４から命令制御部１３に供給される。

【００４０】時刻ｔ＋３では、マルチプル演算命令ａが命令キュー２９から命令キュー３０に供給される。パイプラインド演算命令ｂが命令バッファ２７から命令キュー２９に供給される。命令制御部１３はパイプラインド演算命令ｃを命令バッファ２７に供給する。また、パイプラインド演算命令ｄが命令キャッシュ１４から命令制御部１３に供給される。

【００４１】時刻ｔ＋４では、マルチプル演算命令ａが命令キュー３０からマルチプル演算命令用の命令キュー３１に供給される。パイプラインド演算命令ｂが命令キュー２９から命令キュー３０に供給される。パイプラインド演算命令ｃが命令バッファ２７から命令キュー２９に供給される。命令制御部１３はパイプラインド演算命令ｄを命令バッファ２７に供給する。また、パイプラインド演算命令ｅが命令キャッシュ１４から命令制御部１３に供給される。

【００４２】時刻ｔ＋５では、マルチプル演算命令ａが命令キュー３１に引き続き格納される。そして、パイプラインド演算命令ｂが演算処理を終了し、デキューされる。パイプラインド演算命令ｃが命令キュー２９から命令キュー３０に供給される。パイプラインド演算命令ｄが命令バッファ２７から命令キュー２９に供給される。命令制御部１３はパイプラインド演算命令ｅを命令バッファ２７に供給する。

【００４３】ここで、図2を参照しつつ説明したタイムチャートと比較すると、図2のタイムチャートでは時刻ｔ＋５でＣＰＵストールが発生しているが、図4のタイムチャートでは時刻ｔ＋５でＣＰＵストールを発生していない。つまり、本発明の第１実施例のパイプライン処理装置では、マルチプル演算命令であっても他のパイプラインド演算命令と同様のタイミングで命令キュー２９，３０を遷移する為、後続のパイプラインド命令をストールさせることなく演算処理が可能である。従って、全体としての演算性能が飛躍的に向上すると共に、ストールを極力回避する為に実装されていた複数段の命令バッファの段数を削減することが可能となる。

【００４４】なお、後続の演算命令は先行するマルチプル演算命令が実行中であっても演算例外を発生する場合がある。このように後続の演算命令が演算例外を発生し

た場合、先行する演算命令は実行を完了させるか、マルチプル演算命令の検出時或いは後続の演算命令の演算例外の検出時に、命令キューに演算命令，レジスタ番号，及び演算例外情報等を保持することができる。

【００４５】図５は、本発明のパイプライン処理装置の第２実施例のブロック図を示す。なお、図５のブロック図はパイプライン演算装置のＣＯＰ２０を記載したものであり、ＣＰＵ１０の記載を省略している。また、図３のパイプライン演算装置と同様な部分については同一符号を付して説明を省略している。

【００４６】図５のパイプライン演算装置は、マルチプル演算命令用の命令キュー３７，３８及び演算ステージ３５，３６を設けていることを特徴としている。この場合、マルチプル演算命令用の命令キュー３７，３８及び演算ステージ３５，３６に格納されている演算命令の命令発行順序を外部に通知する必要があり、命令キュー３７，３８にアドレス操作ビット３９，４０を設けることにより、命令発行順序を明確にすることが可能となっている。

【００４７】レイテンシの異なる２つのマルチプル演算命令ａ，ｂを異なるマルチプル演算ステージ３５，３６で実行する場合、この演算ステージ３５，３６に夫々対応するように命令キュー３７，３８及びアドレス操作ビット３９，４０を設ける。アドレス操作ビットとは、命令キューのアドレスを決定する為に使用されるビットである。

【００４８】例えば、命令キュー３７，３８のアドレスに「０００」，「００１」が割り当てられているものとする。マルチプル演算命令ａが発行され、専用の命令キュー３７に格納される際に、他方の命令キュー３８に付随するアドレス操作ビット４０が「０」であればアドレス操作ビット３９に「１」をセットする。また、他方の命令キュー３８に付随するアドレス操作ビット４０が「１」であればアドレス操作ビット３９に「０」をセットする。

【００４９】そして、アドレス操作ビットが「１」であるマルチプル演算命令が実行完了した場合は、そのアドレス操作ビットを「１」から「０」に変更すると共に、他方の命令キューに付随するアドレス操作ビットを「０」から「１」に変更する。つまり、２つのマルチプル演算命令ａ，ｂのうち、先に発行されたマルチプル演算命令用の命令キューに付随するアドレス操作ビットは常に「１」となり、先に発行されたマルチプル演算命令を２ビットのレジスタで知ることが可能である。

【００５０】更に、付随するアドレス操作ビットが「１」であるマルチプル演算命令用の命令キューのアドレスは「０００」であり、付随するアドレス操作ビットが「０」であるマルチプル演算命令用の命令キューのアドレスは「００１」であると決めておくことで、アドレスの昇順に命令キューを読み出せば、常に命令発行順にマルチプル演算命令の情報を命令キューから読み出すことができる。

【００５１】図６は、本発明のパイプライン処理装置の第３実施例のブロック図を示す。なお、図６のブロック図は複数のＣＯＰを有するパイプライン演算装置を記載したものであり、説明に不要な記載を省略している。また、図３のパイプライン演算装置と同様な部分については同一符号を付して説明を省略している。

【００５２】図６のパイプライン処理装置は二つのＣＯＰ５０，６０を有し、夫々のＣＯＰ５０，６０にマルチプル演算命令用の命令キュー５４，６４が含まれる。マルチプル演算命令用の命令キュー５４，６４は、図６の第２実施例と同様に、命令キュー５４，６４に付随するようにアドレス操作ビットが設けられている。

【００５３】このように、パイプライン演算装置に複数のＣＯＰが含まれる場合、異なるＣＯＰに含まれるマルチプル演算命令用の命令キュー５４，６４に同時にマルチプル演算命令が格納されることがある。この場合、マルチプル演算命令が命令キュー５４，６４に同時に発行されたことが保証される限りにおいては、命令キュー５４，６４に付随するアドレス操作ビットに値をセットする優先順位を決めておく必要がある。この優先順位は、ｖａｌｉｄ生成装置５６，６６により決定される。なお、その他のタイミングについては、一つのＣＯＰ内にマルチプル演算命令用の命令キューが実装されている場合と同様であり説明を省略する。

【００５４】

【発明の効果】上述の如く、本発明によれば、マルチプル演算命令を実行する場合に、後続の演算命令に対する命令処理性能の低下を防ぐことが可能であり、命令バッファの段数を削減することにより消費電力及びコストの削減が可能である。

【図面の簡単な説明】

【図１】パイプライン処理装置の一例のブロック図である。

【図２】パイプライン処理装置の一例のタイムチャートである。

【図３】本発明のパイプライン処理装置の第１実施例のブロック図である。

【図４】パイプライン処理装置の一例のタイムチャートである。

【図５】本発明のパイプライン処理装置の第２実施例のブロック図である。

【図６】本発明のパイプライン処理装置の第３実施例のブロック図である。

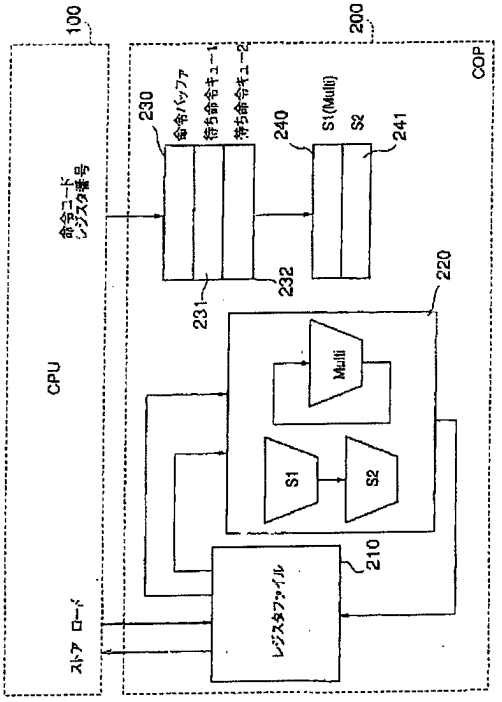
【符号の説明】

１０　中央演算装置
１１　データキャッシュ
１２　整数演算部＆汎用レジスタファイル
１３　命令制御部

14　命令キャッシュ
20　コプロセッサ
21　レジスタファイル
22　演算器
27，51，61　命令バッファ
28　デコーダ
24～26，35，36　演算ステージ
29～31，37，38，52～54，62～64　命令キュー
39，40，55，65　アドレス操作ビット
56，66　valid生成装置

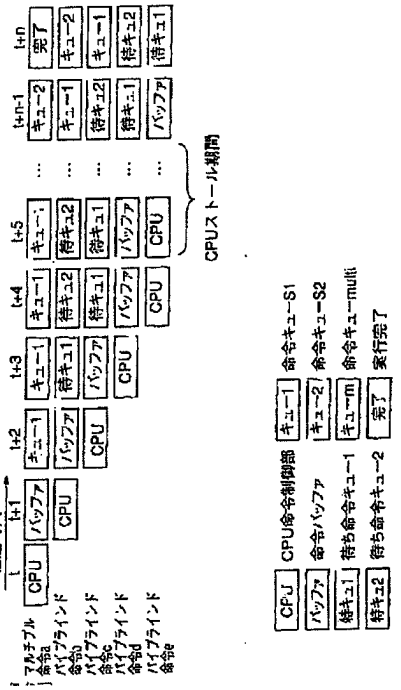
【図1】

パイプライン処理装置の一例のブロック図



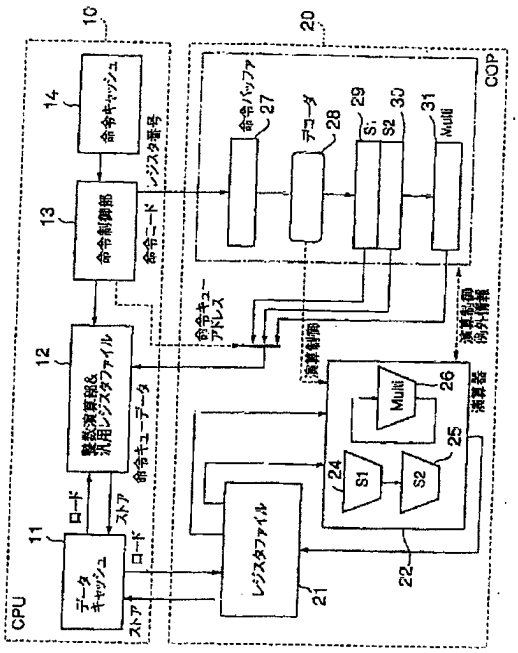

【図2】

パイプライン処理装置の一例のタイムチャート



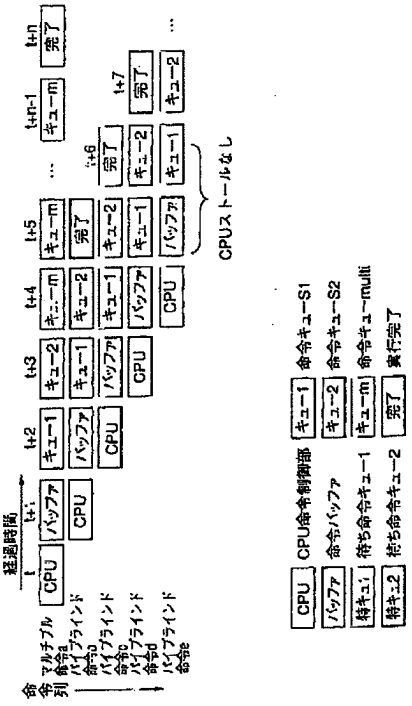
【図3】

本発明のパイプライン処理装置の第1実施例のブロック図



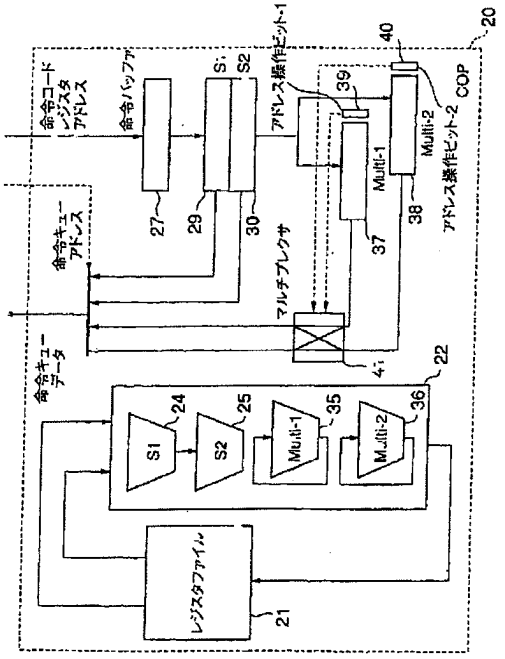

【図4】

パイプライン処理装置の一例のタイムチャート




【図5】

本発明のパイプライン処理装置の第2実施例のブロック図



【図6】

本発明のパイプライン処理装置の第3実施例のブロック図